MŰEGYETEM 1782

Student: **Hamidov Akbar**

Neptun Code: **FM8VEU**

Subject: **Project Lab**

Consultant: **Mrad Mohamed Azouz**

Field of Studying: **Data Science, Machine Learning Applications in non-IT fields**

Department: **Department of Automation and Applied Informatics**

Prediction of In Vitro Dissolution Profile Using Artificial Neural Networks

1. Introduction

Machine Learning (ML) is the field of computer science which is mainly about algorithms those learned from data and train themselves for predictions and making decision without explicitly programmed. ML is known as submodule of Artificial Intelligence (AI). ML is used in different industries as Pharmaceutical Industry. In the following section overview of the project was given.

Abstract: In pharmaceutical industry, dissolution testing is part of the target product quality that are essentials in the approval of new products. The prediction of the dissolution profile based on spectroscopic data is an alternative to the current destructive and time-consuming method. Raman and near infrared (NIR) spectroscopies are two complementary methods, that provide information on the physical and chemical properties of the tablets and can help in predicting their dissolution profiles. This work aims to use the information collected by these methods by creating an artificial neural network model that can predict the dissolution profiles of the scanned tablets. Mathematical dissolution models were used to describe the dissolution profiles, some models were compared and evaluated in this work. The ANN models created used the spectroscopies data along with the measured compression curves as an input to predict the dissolution profiles and the mathematical models' parameters. It was found that ANN models were able to predict the dissolution profile within the acceptance limit of the f1 and f2 factors.[1]

There are several programming languages and libraries for analyzing, visualizing data and train models etc. I have used below technologies on my project:

- Programming Language: Python
- Environment: Jupyter Notebook
- Data Analyzing: NumPy, Pandas
- Data Visualization: Matplotlib, Seaborn
- ML Library: Scikit-learn

There are variety of models (algorithms) which are used for different ML problems. Choosing appropriate algorithm depends on different characteristics such as size of training data, Accuracy and/or Interpretability of the output, speed or training time, linearity, and number of features. Algorithms were used for this problem mentioned below:

- Decision Tree Regressor
- Gradient Boosting Regressor
- Random Forest Regressor
- *Artificial Neural Network*

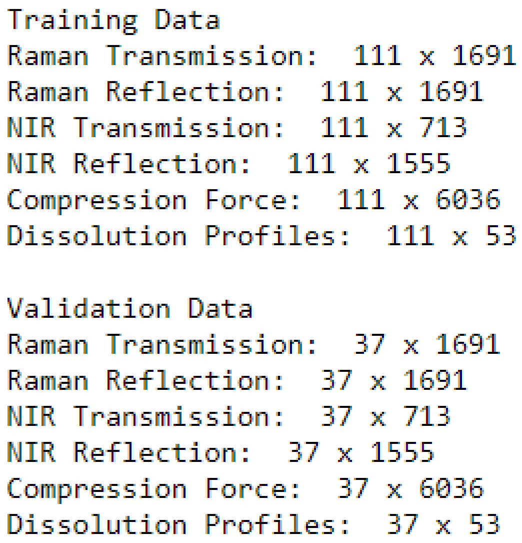
2. Data Analytics

Measurements of the NIR and RAMAN spectroscopy, along with the pressure curves extracted during the compression of the tablets. The data consists of the NIR refection and transmission, Raman refection and transmission spectra, the compression force - time curve and the dissolution profile of 148 tablets. The tablets were produced with a total of 37 different settings. Three parameters were varied: drotaverine content, HPMC content and the compression force. From each setting, four tablets were selected for analysis (37*4). These spectroscopy data results using with given compression force values for different time slot will be used as feature value for model training.

3. Data Pre-processing

As mentioned above, we have several data which are feature values. Target data is dissolution profile in 53 time slots. These data were given in excel files. First, I have split data into train and validation data. For our project given data are those:

- Raman Reflection
- Raman Transmission
- NIR Reflection
- NIR Transmission
- Compression Force
- Dissolution Profiles **(target value)**



```
Training Data
Raman Transmission:  111 x 1691
Raman Reflection:  111 x 1691
NIR Transmission:  111 x 713
NIR Reflection:  111 x 1555
Compression Force:  111 x 6036
Dissolution Profiles:  111 x 53

Validation Data
Raman Transmission:  37 x 1691
Raman Reflection:  37 x 1691
NIR Transmission:  37 x 713
NIR Reflection:  37 x 1555
Compression Force:  37 x 6036
Dissolution Profiles:  37 x 53
```

We have 148 data samples for all mentioned values. I have split 37 of them for validation and rest for train model. All data were read and stored as numPy array, and we can see size of each in the screenshot.

These data values are not sufficient for training data, that is standardization and normalization tools must be applied on feature data. However, we should not apply any of them to Dissolution Profiles, because target values are needed for validation as it is.

3.1 Standard Scaling

Feature Scaling is one of the important pre-processing that is required for standardizing/normalization of the input data. When the range of values are very distinct in each column, we need to scale them to the common level. Feature Scaling is required for correct prediction and results**.** In case when the values of one of the column is very high as compared to others, the impact of the column with higher value will be much higher as compared to the impact of other low valued columns**.** The feature with high magnitude will weigh lot more than features having low magnitude even if they are more crucial in determining the output[2].

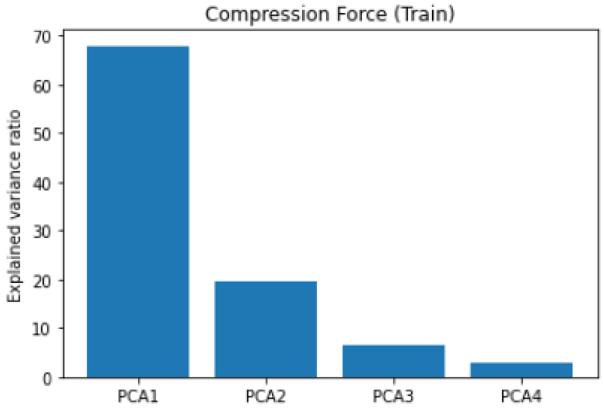
3.2 Principal Component Analysis (PCA)

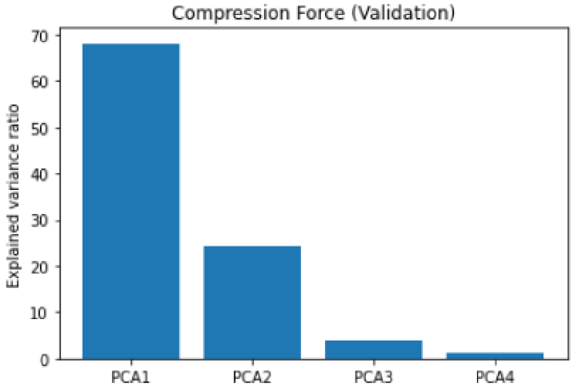
Principal Component Analysis, or PCA, is a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set.

Reducing the number of variables of a data set naturally comes at the expense of accuracy, but the trick in dimensionality reduction is to trade a little accuracy for simplicity. Because smaller data sets are easier to explore and visualize and make analyzing data much easier and faster for machine learning algorithms without extraneous variables to process[3].

After applying PCA to feature data, I have visualized efficiency of PCA components.

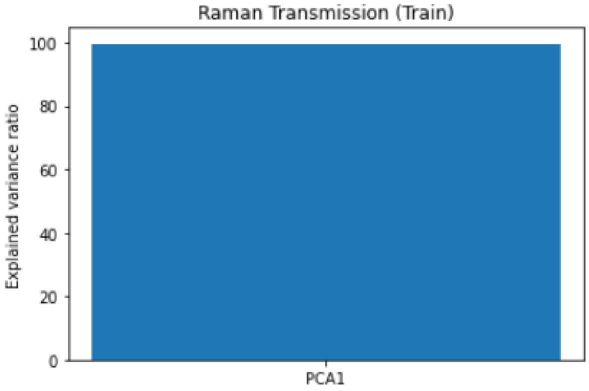
- Compression Force

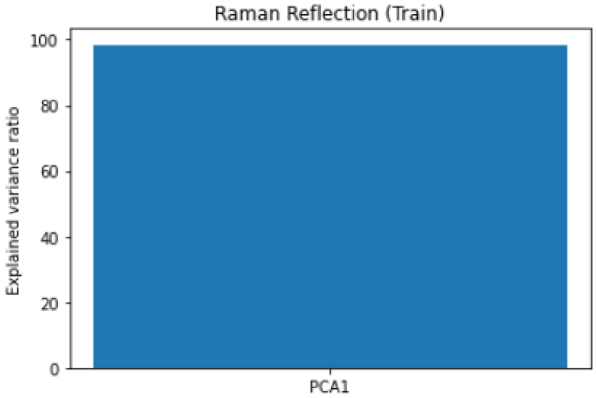


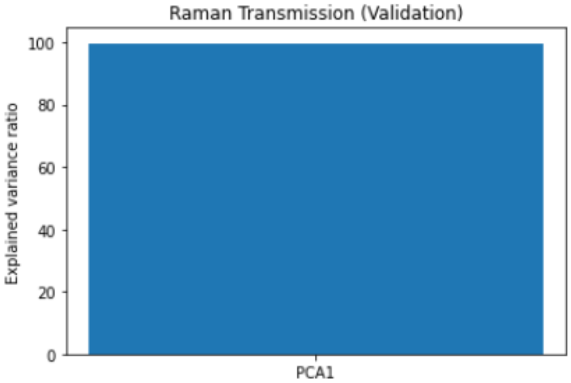


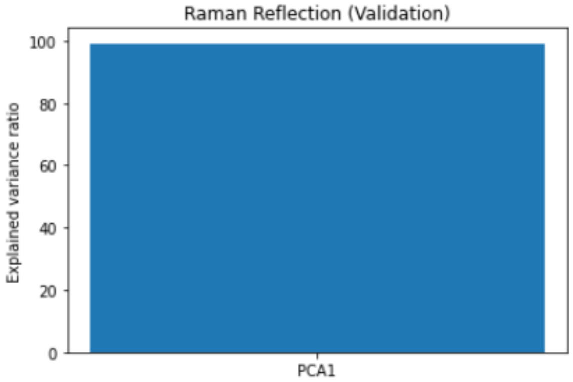
Note: *Compression force values were reduced to 4-dimension because of getting better accuracy (efficiency). In following sections maximum value of compression force were also used instead of these PCA components, because in some algorithms this way gave better results.*

- Raman Transmission and Reflection (Train and Validation)



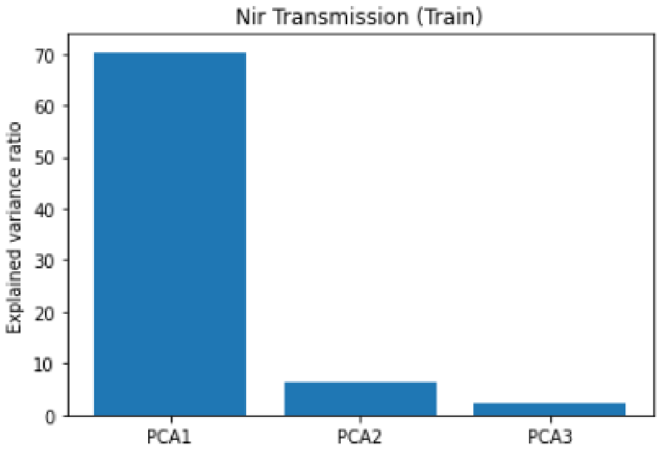


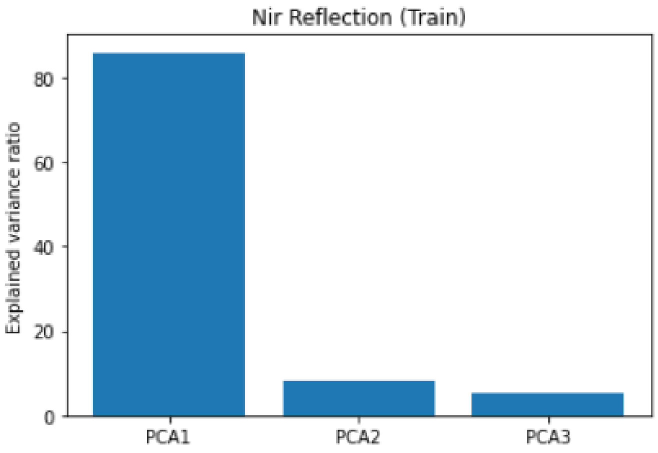


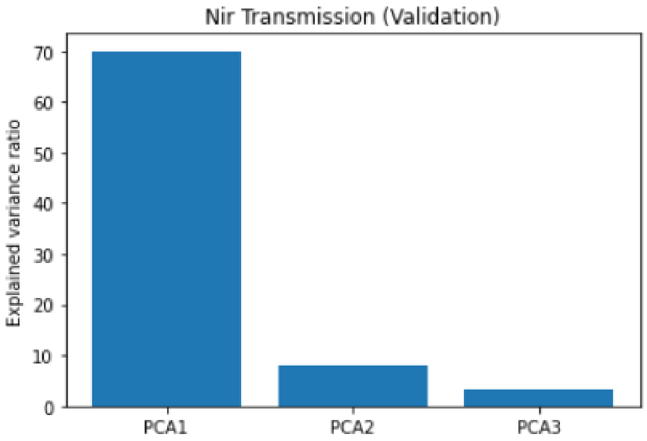


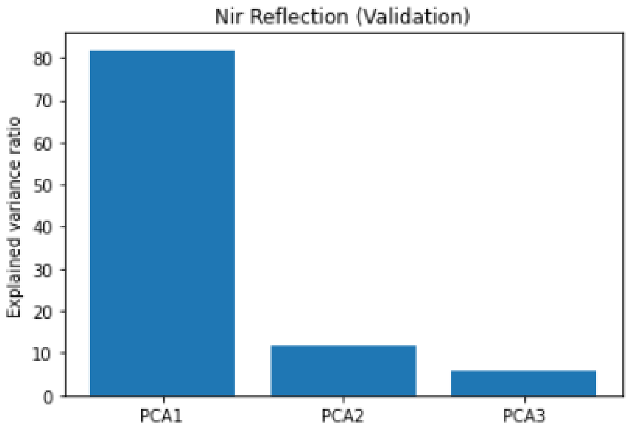
Note: *Raman values were reduced to 1-dimension because it gives enough higher efficiency. That is why each graph corresponds to proper data which labeled, that means each Raman datasheet has one PCA component.*

- NIR Transmission and Reflection

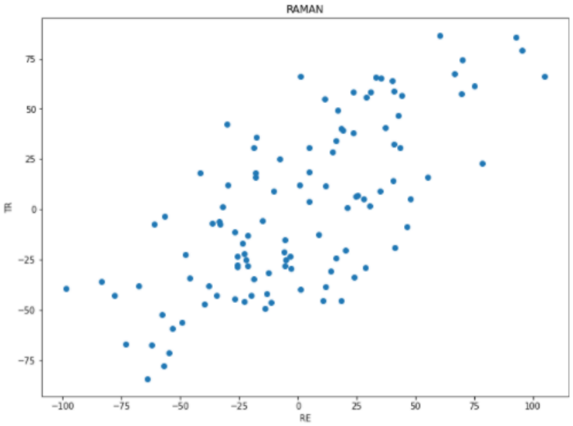


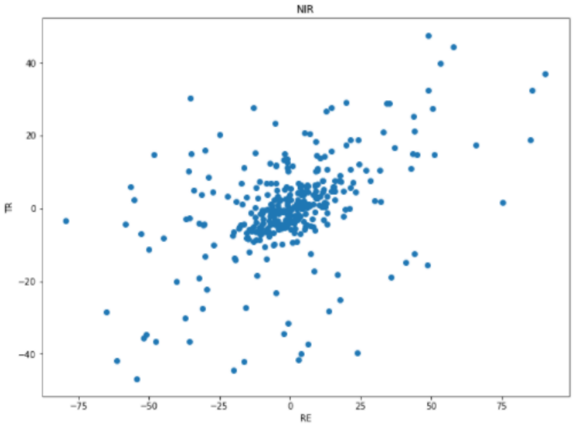






I have visualized relationship between reflection and transmission values of NIR/RAMAN values. I have plotted them using scatter plot and by this I can see Raman values are distributed better than Nir values.

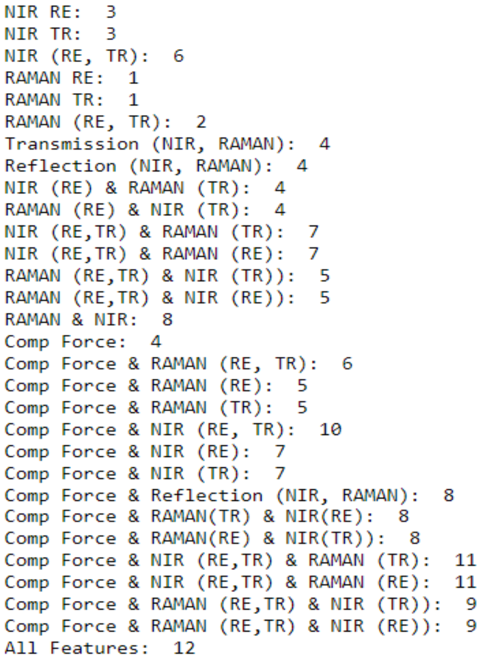




4. Data Combinations

After data cleaning, there are 5 train and 5 validation matrices, and one dissolution profiles matrix for train data and another one for validation data. Each train data matrix 111 and validation matrix has rows and corresponding number of columns due to resulting PCA components. Each matrix is numPy array.

I have created 30 different possible feature matrices for checking efficiency of models and determining which combination can be understandable by algorithm (model). Combination references and their number of columns after concatenation of referred arrays can be seen in screenshot. There is one more data combination besides these 30 ones which referred as all features that means all possible feature arrays are concatenated.



```
NIR RE:  3
NIR TR:  3
NIR (RE, TR):  6
RAMAN RE:  1
RAMAN TR:  1
RAMAN (RE, TR):  2
Transmission (NIR, RAMAN):  4
Reflection (NIR, RAMAN):  4
NIR (RE) & RAMAN (TR):  4
RAMAN (RE) & NIR (TR):  4
NIR (RE,TR) & RAMAN (TR):  7
NIR (RE,TR) & RAMAN (RE):  7
RAMAN (RE,TR) & NIR (TR)):  5
RAMAN (RE,TR) & NIR (RE)):  5
RAMAN & NIR:  8
Comp Force:  4
Comp Force & RAMAN (RE, TR):  6
Comp Force & RAMAN (RE):  5
Comp Force & RAMAN (TR):  5
Comp Force & NIR (RE, TR):  10
Comp Force & NIR (RE):  7
Comp Force & NIR (TR):  7
Comp Force & Reflection (NIR, RAMAN):  8
Comp Force & RAMAN(TR) & NIR(RE):  8
Comp Force & RAMAN(RE) & NIR(TR)):  8
Comp Force & NIR (RE,TR) & RAMAN (TR):  11
Comp Force & NIR (RE,TR) & RAMAN (RE):  11
Comp Force & RAMAN (RE,TR) & NIR (TR)):  9
Comp Force & RAMAN (RE,TR) & NIR (RE)):  9
All Features:  12
```

5. Used Algorithms (Models)

As mentioned in introduction, I have used 3 different algorithms (Artificial Neural Networks was not considered as model here). Each model was trained in 30 iteration with each possible data combination and f2 function was calculated after each iteration, in the end percentage of efficiency out of 100 for this model in this iteration was stored.

Results those under 50% are not considered usable model training. Due to fact, dissolution profiles for both train and validation are same to all iteration and we can pick any identical dissolution profile value and check model's estimation in visual format. For this reason, 53 time slot were given, and likelihood of real dissolution profile curve and model's estimation was visualized.

I have used only regressor algorithms because after research on characteristics of this problem I have end up using these 3 regressor models.

Article. *Regression algorithms fall under the family of Supervised Machine Learning algorithms which is a subset of machine learning algorithms. One of the main features of supervised learning algorithms is that they model dependencies and relationships between the target output and input features to predict the value for new data. Regression algorithms predict the output values based on input features from the data fed in the system. The go-to methodology is the algorithm builds a model on the features of training data and using the model to predict the value for new data.* [4]

5.1 Decision Tree Regressor

A decision tree is arriving at an estimate by asking a series of questions to the data, each question narrowing our possible values until the model get confident enough to make a single prediction. The order of the question as well as their content are being determined by the model. In addition, the questions asked are all in a True/False form.[5]

5.2 Random Forest Regressor

A Random Forest is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap and Aggregation, commonly known as bagging. The basic idea behind this is to combine multiple decision trees in determining the final output rather than relying on individual decision trees. Random Forest has multiple decision trees as base learning models. We randomly perform row sampling and feature sampling from the dataset forming sample datasets for every model. This part is called Bootstrap.[6]

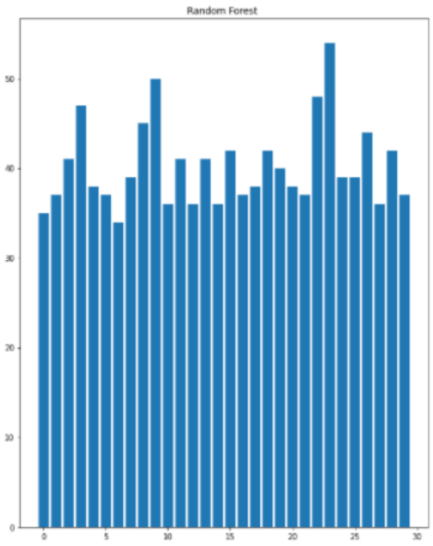
5.3 Gradient Boosting Regressor

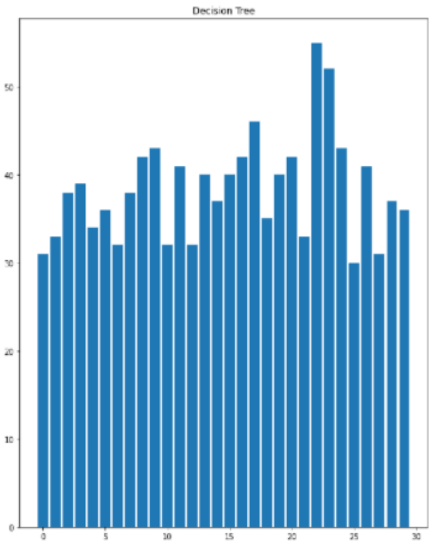
Gradient boosting is a type of machine learning boosting. It relies on the intuition that the best possible next model, when combined with previous models, minimizes the overall prediction error. The key idea is to set the target outcomes for this next model to minimize the error.[7]

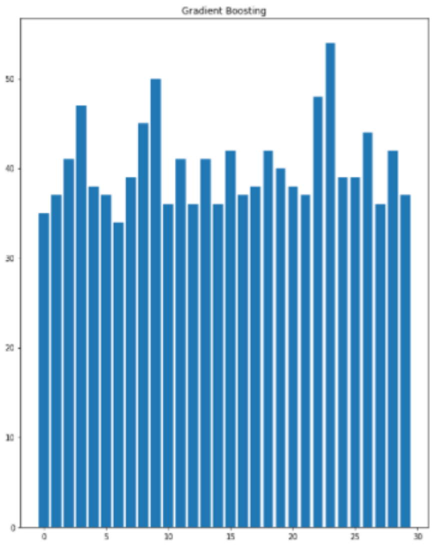
6. Statistical Approach

I have trained each model for each data combination and recorded their estimation efficiency. Two data combinations are resulted better for all three algorithms. First combination was Raman reflection, Nir reflection and Compression Force values and the second one was Raman reflection, Nir Transmission and Compression force values.

In other data combinations, there were some useful (better resulted) iterations too. But these two combinations work for all algorithms. In the graphs below we can see visualization of results for each algorithm.







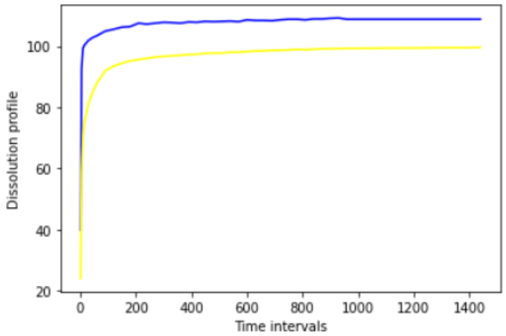
As shown graph only few results are higher than 50% those can be used to estimate dissolution profiles. By looking at these useful (good resulted) results we observe see data those were concatenated with compression force are better. From those observations, I have come to idea that compression force is more meaningful for models. As given previous sections, compression force data has 6036 columns in original. After applying PCA I checked all iterations and noted them down. Then, I have only used maximum value for each pill test from compression force values which resulted slightly better.

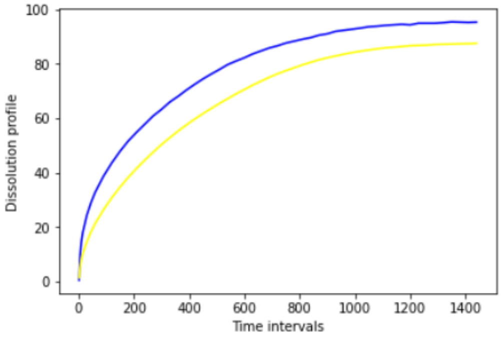
7. Dissolution Curves

After getting some useful iterations, I have plotted original and estimated dissolution curves for comparing efficiency. For each iteration I have added two dissolution curve for visualizing their comparison.

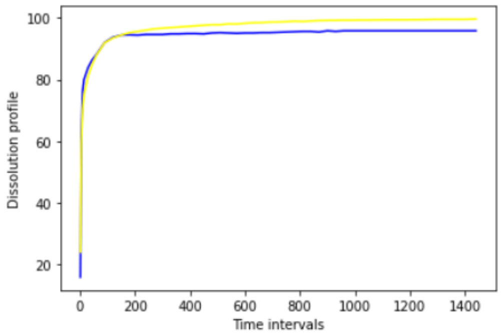
Note: *Dissolution curves were shown are only from iterations which resulted above 50%.*

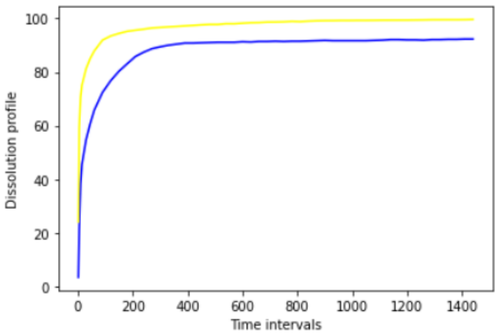
- **Decision Tree Regressor**
  - Train Data: Raman Reflection, NIR Reflection and Compression Force
  - F2 function result: 55%



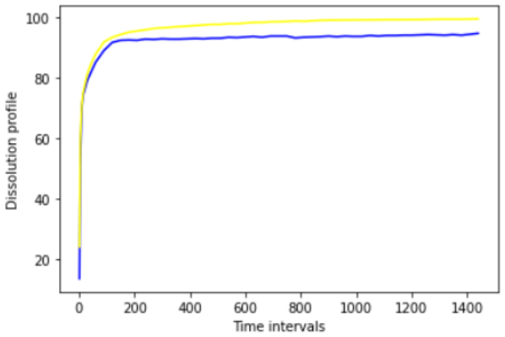


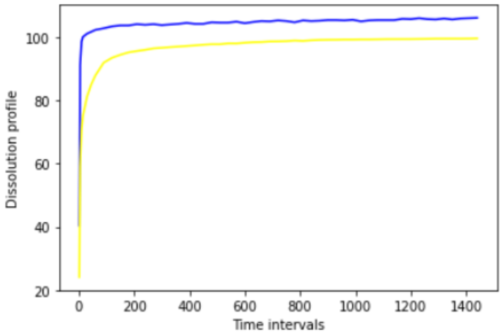
- **Decision Tree Regressor**
  - Train Data: Raman Reflection, NIR Transmission and Compression Force
  - F2 function result: 52%



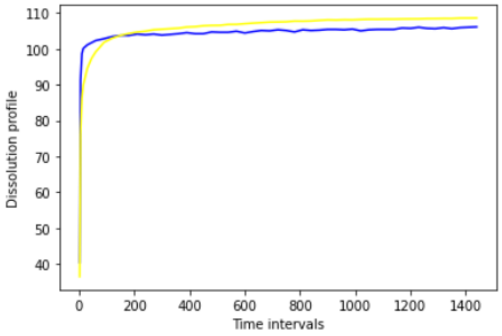


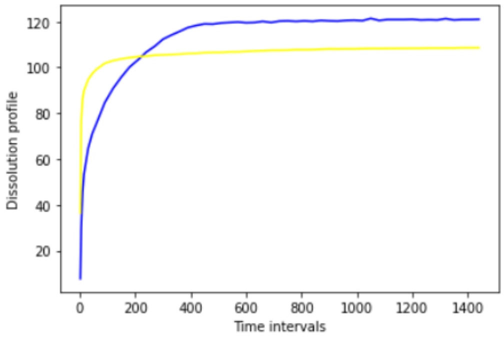
- **Random Forest Regressor**
  - Train Data: Raman Reflection, NIR Reflection
  - F2 function result: 50%



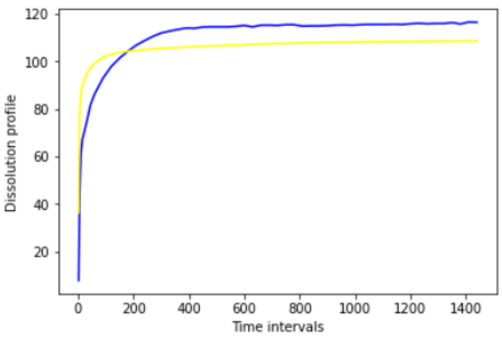


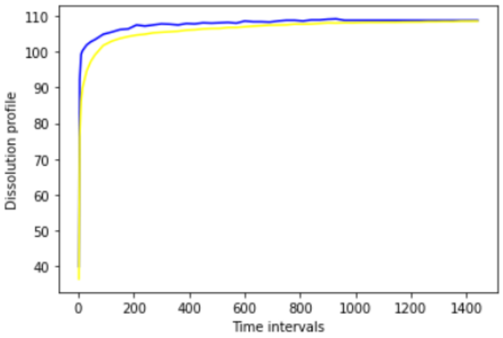
- **Random Forest Regressor**
  - Train Data: Raman Reflection, NIR Transmission and Compression Force
  - F2 function result: 54%

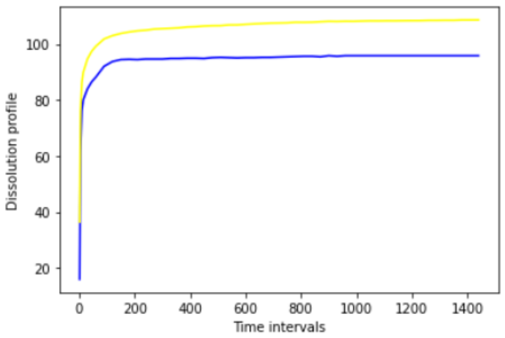




- **Gradient Boosting Regressor**
  - Train Data: Raman Reflection, NIR Transmission and Compression Force
    - F2 function result: 59%
  - Train Data: Raman Reflection, NIR Reflection and Compression Force
    - F2 function result: 51%







8. Conclusion

As we can see models trained themselves due to different logics and giving different results. In some bad resulted iterations, I have also visualized dissolution curves. In that case, I observe that in most cases overfitting occurs due to noisy data, that is why models are not able estimate dissolution curves correctly.

In statistics, overfitting is "the production of an analysis that corresponds too closely or exactly to a particular set of data and may therefore fail to fit additional data or predict future observations reliably".[8]

For eliminating ‘overfitting’ there are several Machine Learning Techniques that can be applied such as: cross-validation, regularization, assembling. One more cause for this is amount of train data. We have only 111 train data and this cannot be enough for models in most cases. Of course, more data does not mean good training always, but this also can be powerful reason.

Lastly, I have built Artificial Neural Network for getting better results, however I got only around 45%. After researching a bit more, I realized clustering is needed in advance for training models for this problem. In the future, collecting more data and clustering them and building Neural Networks with 3-4 layer (input and output layers included) and each layer can have 2-4 neurons can be resulted better. Additionally, for data cleaning normalization can be used instead of scaling values.

References

[1] Mohamed Azouz Mrad, Kristóf Csorba Dorián László Galata, Zsombor Kristóf Nagy, Brigitta Nagy “Prediction of In Vitro Dissolution Profile Using Artificial Neural Networks.”

[2] https://medium.com/technofunnel/what-when-why-feature-scaling-for-machine-learning-standard-minmax-scaler-49e64c510422

[3] https://builtin.com/data-science/step-step-explanation-principal-component-analysis

[4] https://analyticsindiamag.com/top-6-regression-algorithms-used-data-mining-applications-industry/#:~:text=Regression%20algorithms%20fall%20under%20the,subset%20of%20machine%20learning%20algorithms.&text=Regression%20algorithms%20predict%20the%20output,data%20fed%20in%20the%20system.

[5] https://gdcoder.com/decision-tree-regressor-explained-in-depth/

[6] https://towardsdatascience.com/random-forests-an-ensemble-of-decision-trees-37a003084c6c

[7] https://www.displayr.com/gradient-boosting-the-coolest-kid-on-the-machine-learning-block/

[8] https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms